PATENT

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE
BOARD OF PATENT APPEALS AND INTERFERENCES

In Re Application of:	)	
	)	
Srivatsa Krishnaswamy, et al.	)	Confirmation No. 5027
	)	
Serial No.: 10/808,223	)	Examiner: Zhen, Li B.
	)	Group Art Unit: 2194
Filed: March 23, 2004	)	
	)	
For: **Method and System for Data Object Transformation**	)	HP Docket No.: 200300248-1
	)	

APPEAL BRIEF UNDER 37 C.F.R. § 41.37

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

Sir:

This Appeal Brief is submitted in support of the Notice of Appeal filed herewith, responding to the final Office Action mailed June 3, 2009.

REAL PARTY IN INTEREST

The real party in interest of the instant application is Hewlett-Packard Development Company, a Texas Limited Liability Partnership having its principal place of business in Houston, Texas.

I. RELATED APPEALS AND INTERFERENCES

There are no related appeals or interferences.

II. STATUS OF THE CLAIMS

Claims 1-6, 11-13 and 15-26 are pending in this application. Claims 1-6, 11-13 and 15-26 were rejected by the final Office Action, and are the subject of this appeal. Claims 7-10 and 14 were cancelled during prosecution.

III. STATUS OF AMENDMENTS

There have been no claim amendments made after the final Office Action, and all amendments made before the final Office Action have been entered. The claim listing in section VII (CLAIMS – APPENDIX) represents the present state of the claims.

IV. SUMMARY OF THE CLAIMED SUBJECT MATTER

Embodiments of the claimed subject matter are summarized below with reference numbers and references to the written description ("specification") and drawings. The subject matter described below appears in the original disclosure at least where indicated, and may further appear in other places within the original disclosure.

Embodiments according to independent claim 1 involve a method of data object transformation between a middleware and a application, the method comprising: receiving a message from a messaging middleware (p. 17, line 18 – p. 18, line 9; FIG. 3: 78) by a data transformation adapter (p. 18, lines 4-9; FIG. 3: 78), the message including one or more data objects in an eXtensible Markup Language (XML) (p. 18, lines 17-29), wherein the message is a first communications format (p. 19, lines 4-5); converting by the data transformation adapter the message from the first communications format to a second communications format (p. 18, lines 4-9; FIG. 3: 78, 80, 82); converting by the data transformation adapter the one or more data objects in

XML to a non-eXtensible Markup Language (non-XML) (p. 13, line 10 – p. 14, line 21; FIG. 3, 88) wherein the one or more data objects are converted using a first set of one or more transformation classes, the one or more transformation classes being configured to transform the one or more data objects in XML to non-XML (p. 13, line 10 – p. 14, line 21; FIG. 3, 88), each of the one or more transformation classes generated using mapping rules, the mapping rules including XML based syntax that uses rule specification guide to facilitate transforming the one or more data objects in XML to non-XML (p. 13, line 10 – p. 14, line 21); and transmitting by the data transformation adapter the one or more data objects in non-XML to an application (p. 18, lines 10-16; FIG. 3: 92).

Embodiments according to claim 11 involve a data transformation adapter having program instructions stored in memory (FIG. 5: 220), the program instructions comprising: generating a first object model and a second object model, the first object model including a plurality of data objects in an eXtensible Markup Language (XML), and the second object model including a plurality of data objects in a non-eXtensible Markup Language (non-XML) (p. 13, line 10 – p. 14, line 21; FIG. 3, 88); storing the first and second object models in one or more memories (FIG. 3: 88; FIG. 5: 220); generating mapping rules, the mapping rules including XML based syntax that uses rule specification guide to facilitate transforming the one or more data objects in XML to non-XML (p. 13, line 10 – p. 14, line 21); generating a plurality of transformation classes using the first and second object models and the transformation mapping rules, the one or more transformation classes being configured to transform the one or more data objects in XML to non-XML (p. 13, line 10 – p. 14, line 21; FIG. 3, 88); receiving one or

more data objects; converting the received one or more data objects (p. 18, lines 4-16; FIG. 3: 88, 90), via the transformation classes, (1) in XML to non-XML; or (2) in non-XML to XML (p. 13, line 10 – p. 14, line 21; FIG. 3, 88); and transmitting the converted one or more data objects (FIG. 3: 92).

Embodiments according to claim 17 involve a system for data object transformation, the system comprising one or more processors (FIG. 5: 200); one or more memories coupled to the one or more processors (FIG. 5: 220); and a data transformation adapter having program instructions stored in the one or more memories (FIG. 4: 134), the one or more processors being operable to execute the program instructions, the program instructions including: receiving a message from a messaging middleware, the message including one or more data objects in an eXtensible Markup Language (XML) (p. 13, line 10 – p. 14, line 21; FIG. 3, 88), wherein the message is in a first communications format (p. 19, lines 4-5); converting the message from the first communications format to a second communications format (p. 18, lines 4-9; FIG. 3: 78, 80, 82); converting the one or more data objects in XML to a non-eXtensible Markup Language (non-XML) (p. 13, line 10 – p. 14, line 21; FIG. 3, 88), wherein the one or more data objects are converted using a first set of one or more transformation classes, the one or more transformation classes being configured to transform the one or more data objects in XML to non-XML, each of the one or more transformation classes generated using mapping rules , the mapping rules including XML based syntax that uses rule specification guide to facilitate transforming the one or more data objects in XML to non-XML (p. 13, line 10 – p. 14, line 21); and transmitting the one or more data objects in non-XML to an application (FIG. 3: 92).

Embodiments according to claim 22 involve a system for data object transformation, the system comprising: a communications line (p. 20, line 30 – p. 21, line 14); a computer readable medium executable on a computing system (FIG. 5), the computing system coupled to the communications line (FIG. 5: 240), the computer readable medium having a transformation adapter (FIG. 4: 114), the transformation adapter including: an assembly/disassembly layer configured to convert messages from a first communications format to a second communications format (p. 18, lines 4-9; FIG. 3: 78, 80, 82); a transformation layer configured to convert data objects in an eXtensible Markup Language (XML) to a non-eXtensible Markup Language (non-XML) using one or more transformation classes (p. 13, line 10 – p. 14, line 21), the one or more transformation classes being configured to transform the one or more data objects in XML to non-XML (p. 18, lines 4-9; FIG. 3: 78, 80, 82); and a method invocation layer; a transformation class generator coupled to the transformation adapter, the transformation class generator configured to generate the one or more transformation classes using transformation mapping rules (FIG. 4: 114), the mapping rules including XML based syntax that uses rule specification guide to facilitate transforming the one or more data objects in XML to non-XML (p. 18, lines 4-9; FIG. 3: 78, 80, 82); and an application coupled to the transformation adapter, wherein the application transmits data to and receives data from the method invocation layer (FIG. 3: 92).

Embodiments according to claim 26 involve an apparatus for data object transformation, the apparatus comprising: means for generating a first object model and a second object model, the first object model including a plurality of data objects in an eXtensible Markup Language (XML), and the second object model including a plurality

of data objects in a non-eXtensible Markup Language (non-XML) (p. 13, line 10 – p. 14, line 21; FIG. 3, 88); means for storing the first and second object models (FIG. 3: 88; FIG. 5: 220); means for generating transformation mapping rules, the mapping rules including XML based syntax that uses rule specification guide to facilitate transforming the one or more data objects in XML to non-XML (p. 13, line 10 – p. 14, line 21); means for generating a plurality of transformation classes using the first and second object models and the transformation mapping rules, the one or more transformation classes being configured to transform the one or more data objects in XML to non-XML (p. 13, line 10 – p. 14, line 21; FIG. 3, 88); means for receiving one or more data objects; means for converting the received data objects (p. 18, lines 4-16; FIG. 3: 88, 90), via the transformation classes, in XML to non-XML (p. 13, line 10 – p. 14, line 21; FIG. 3, 88); and means for transmitting the converted one or more data objects (FIG. 3: 92).

V. GROUNDS OF REJECTION TO BE REVIEWED ON APPEAL

The following grounds of rejection are to be reviewed on appeal.

A. Claims 1-6, 11-13 and 15-26 stand rejected under 35 U.S.C. § 112, first paragraph, as allegedly failing to comply with the written description requirement.

B. Claims 22-25 were rejected under 35 U.S.C. § 101 as allegedly directed to non-statutory subject matter.

C. Claims 1-6, 11-13 and 15-26 stand rejected under 35 U.S.C. § 103(a) as allegedly being unpatentable over U.S. Patent Application Publication No. 2002/0120652 to Rising et al. (hereinafter "*Rising*") in view of U.S. Patent Application Publication No. 2003/0037174 to Lavin et al (hereinafter "*Lavin*").

VI. ARGUMENT

A. Rejection of claims 1-6, 11-13 and 15-26 under 35 U.S.C. §112, first paragraph

In item 5 of the Office action, claims 1-6, 11-13 and 15-26 have been rejected under 35 U.S.C. § 112, first paragraph, as allegedly failing to comply with the written description requirement. The Office action alleges that there does not appear to be a specific disclosure of converting data in XML to non-XML. Applicants respectfully disagree. As one non-limiting example, page 13, line 10 – page 14, line 21 describes exemplary code for ***target object creation***. In other words, the claimed invention describes transforming object data in XML to non-XML target objects. The non-limiting exemplary code noted above discloses a specification of a process to transform a source object that can be represented by XML data to a target object that can be represented by non-XML data, such as in a Java object, or an object represented by another programming language. The depicted XML code specifies various fields and syntax for such transformation.

As further described in the specification, the exemplary code can be employed to generate a transformation class that can apply an "expression on source data elements and [assign] them to the target object." In the example contained in the above noted portion of the specification, a source object is mapped into a target object array that can be represented in a non-XML programming language. A transformation class performing the transformation can be subsequently compiled to improve the efficiency of data transformation. Accordingly, Applicants respectfully submit that above noted allegation of the Office action is misplaced, and that the rejection under 35 USC § 112 should be withdrawn.

The Office Action further alleges that dependent claims 4, 13, 20, and 25 identify non-XML application-specific object model types but that the specification discloses application-specific object model types as XML on page 8, lines 8-9 and line 27 of the specification. Applicants again respectfully disagree. The cited portion of Applicants' specification describes a non-limiting ***XML schema*** of an application-specific object model type. Further examination of the disclosed XML schema reveals that it expresses the structure and constraints of a ***non-XML*** application-specific object model type. For example, lines 35-37 of page 8 of the specification discloses certain application-specific constrants of the non-XML object (*e.g.*, element name is "Id" and element type is "integer.") Accordingly, Applicants respectfully request that the rejection of the claims under 35 USC § 112 in this regard also be withdrawn.

B. **Rejection of claims 22-25 under 35 U.S.C. §101**

In item 7 of the Office action, claims 22-25 have been rejected under 35 U.S.C. § 101 because the claimed invention is allegedly directed to non-statutory subject matter. The final Office Action alleges in item 7 that the "specification does not provide 'antecedent basis' for the term computer readable medium." The Office Action also conflates a computer readable medium with a transmission medium, which does not appear in the claim language. Applicant respectfully disagrees with the contentions of the Office Action in this regard. MPEP 2173.05(e) clearly provides the following:

> "A CLAIM TERM WHICH HAS NO ANTECEDENT BASIS IN THE DISCLOSURE IS NOT NECESSARILY INDEFINITE
>
> The mere fact that a term or phrase used in the claim has no antecedent basis in the specification disclosure does not mean, necessarily, that the term or phrase is indefinite. There is no requirement that the words in the

> claim must match those used in the specification disclosure. Applicants are given a great deal of latitude in how they choose to define their invention so long as the terms and phrases used define the invention with a reasonable degree of clarity and precision."

Accordingly, Applicants respectfully submit that support for a computer readable medium can be found on at least page 20, lines 18-24 of the specification, which recites the following:

> Computer program instructions for implementing the network inventory adapter may be stored on the disk storage device 225 until the processor 200 retrieves the computer program instructions, either in full or in part, and stores them in main memory 220. The processor 200 then executes the computer program instructions stored in the main memory 220 to implement the features of network inventory adapter. The program instructions may be executed with a multiprocessor computer having more than one processor.

Accordingly, Applicants submit that the specification provides ample support for a computer readable medium and that the rejection is misplaced and should be overturned.

C. Rejection of claims 1-6, 11-13 and 15-26 under 35 U.S.C. §103(a): *Rising* and *Lavin*

In item 10 of the Office action, claims 1-6, 11-13, and 15-26 have been rejected under 35 U.S.C. § 103(a) as allegedly being unpatentable over U.S. Patent Application Publication No. 2002/0120652 filed by Rising et al. (hereafter "*Rising*") in view of U.S. Patent Application Publication No. 2003/0037174 filed by Lavin et al. (hereafter "*Lavin*"). A prima facie case of obviousness is established only when the prior art teaches or suggests all of the elements of the claims. MPEP §2143.03, In re Rijckaert, 9 F.3d 1531, 28 U.S.P.Q2d 1955, 1956 (Fed. Cir. 1993).

1. **Independent Claim 1**

Claim 1 (with emphasis added) recites:

> 1. A method of data object transformation between a middleware and a application, the method comprising:
> receiving a message from a messaging middleware by a data transformation adapter, the message including one or more data objects in an eXtensible Markup Language (XML), wherein the message is a first communications format;
> converting by the data transformation adapter the message from the first communications format to a second communications format;
> ***converting by the data transformation adapter the one or more data objects in XML to a non-eXtensible Markup Language (non-XML)***, wherein the one or more data objects are converted using a first set of one or more transformation classes, the one or more transformation classes being configured to transform the one or more data objects in XML to non-XML, each of the one or more transformation classes generated using mapping rules, ***the mapping rules including XML based syntax that uses rule specification guide to facilitate transforming the one or more data objects in XML to non-XML***; and
> transmitting by the data transformation adapter the one or more data objects in non-XML to an application.

Applicants respectfully submit that *Rising* in view of *Lavin* fails to disclose at least the above emphasized elements of claim 1. Applicants submit that the cited *Rising* reference discloses achieving compression of an instance document by associating elements and attributes of an instance document to various tokens and/or codes to encode an additional instance document. *See* paragraph 27, *Rising*. This is accomplished via XSLT transformation as noted in paragraph 32 of the reference. Paragraph 32 states the following:

> **An entity, such as server 106, transforms the DDL instance document into an appropriate ASDL instance document <u>using the published XSLT document</u>** and binarizes the ASDL instance document using the published ASDL frequency tables.

In addition, Rising discloses "publishing an XSLT document containing the transformation functions for mapping into the application specific markup language, and

publishing the frequency tables for the ASDL namespace for access by the clients 102, 104 over a communications network 110." In contrast, claim 1 discloses employing ***mapping rules including XML based syntax that uses rule specification guide to facilitate transforming the one or more data objects in XML to non-XML.*** In other words, the mapping rules facilitate transformation of the data objects to non-XML data without resorting to XSLT transformation and its attendant inefficiencies (*e.g.* due to xpath queries).

Accordingly, a solution according to *Rising* relies on XSLT transformation for transforming a first document into a second document to achieve an overall solution, as a server generating an instance document according to *Rising* must still employ XSLT transformation. However, as previously noted, the claimed invention is an alternative solution for XSLT. As noted in the Background of the application, "XSLT and other existing methods have not fully addressed the issues relating to data transformation in a communications network environment. Accordingly, there remains a need for a method and system device that solves existing shortcoming relating to data transformation." The claimed invention increases performance by transforming object data in XML to non-XML rather than to an XML format as a result of an XSLT transformation. The object data in non-XML can be used in an application programming interface and does not involve run-time interpretation of transformation specification as in XSLT transformation. In contrast to *Rising*, the claimed invention improves the performance of data transformation compared with XSLT transformation by removing XSLT transformation from the data transformation process.

Applicants respectfully assert that *Rising,* like references cited by previous Office actions in the instant application, focuses on a system that uses XSLT transformation to achieve an overall solution. The XSLT transformation transforms a first XML document type into a second XML document type. *Lavin* fails to remedy these deficiencies. Therefore, Applicants respectfully submit that *Rising* in view of *Lavin* fail to teach, disclose, or suggest ***transforming the data objects in XML to non-XML***, as recited in claim 1. Consequently, for at least this reason, among others, Applicants respectfully request that claim 1 be allowed and the rejection be withdrawn.

2. Independent claims 11, 17, 22 and 26

Applicants respectfully submit that for reasons related to those set forth above, *Rising* in view of *Lavin* fails to teach, disclose, or suggest each and every element of independent claims 11, 17, 22 and 26.

3. Independent claims 1, 11, 17, 22, and 26

Because independent claims 1, 11, 17, 22 and 26 are allowable over the cited art of record, dependent claims 2-6, 12-16, 18-21, and 23-25 are allowable as a matter of law for at least the reason that dependent claims 2-6, 12-16, 18-21, and 23-25 contain all features and elements of their respective independent base claims. *In re Fine*, 837 F.2d 1071, 5 U.S.P.Q.2d 1596, 1600 (Fed. Cir. 1988). Accordingly, the rejection to dependent claims 2-6, 12-16, 18-21, and 23-25 should be withdrawn for at least this reason, among others.

CONCLUSION

For at least the reasons discussed above, Appellant respectfully requests that the Examiner's final rejection of claims 1-2, 4-13, 15 and 17-25 be overturned by the Board. In addition to the claims listed in Section VII (CLAIMS – APPENDIX), Section VIII (EVIDENCE – APPENDIX) included herein indicates that there is no additional evidence relied upon by this brief. Section IX (RELATED PROCEEDINGS – APPENDIX) included herein indicates that there are no related proceedings.

Respectfully submitted,

By: /arr/

Arvind R. Reddy
Reg. No. 63,007

THOMAS, KAYDEN, HORSTEMEYER & RISLEY, L.L.P.

600 Galleria Parkway, NW
Suite 1500
Atlanta, Georgia 30339-5948
Tel: (770) 933-9500
Fax: (770) 951-0933

VII. CLAIMS – APPENDIX

1. A method of data object transformation between a middleware and a application, the method comprising:

receiving a message from a messaging middleware by a data transformation adapter, the message including one or more data objects in an eXtensible Markup Language (XML), wherein the message is a first communications format;

converting by the data transformation adapter the message from the first communications format to a second communications format;

converting by the data transformation adapter the one or more data objects in XML to a non-eXtensible Markup Language (non-XML), wherein the one or more data objects are converted using a first set of one or more transformation classes, the one or more transformation classes being configured to transform the one or more data objects in XML to non-XML, each of the one or more transformation classes generated using mapping rules, the mapping rules including XML based syntax that uses rule specification guide to facilitate transforming the one or more data objects in XML to non-XML; and

transmitting by the data transformation adapter the one or more data objects in non-XML to an application.

2. The method according to claim 1, wherein the first communications format includes a middleware-dependent format, and the second communications format includes a middleware-independent format.

3. The method according to claim 1, wherein each of the one or more data objects includes a Java object.

4. The method according to claim 1, wherein the XML includes a domain object model type and the non-XML includes an application-specific object model type.

5. The method according to claim 1, further comprising:
registering the application with the messaging middleware; and transmitting high-level function calls to the application.

6. The method according to claim 1, the method further comprising:

receiving a second message from the application, the second message including one or more data objects in non-XML;

converting the one or more data objects in non-XML to XML, wherein the one or more data objects are converted using a second set of one or more of the transformation classes;

generating a communications line dependent message, the communications line dependent message including the one or more data objects in XML; and

transmitting the communications line dependent message to the messaging middleware.

7-10. (Cancelled)

11. A data transformation adapter having program instructions stored in memory, the program instructions comprising:

generating a first object model and a second object model, the first object model including a plurality of data objects in an eXtensible Markup Language (XML), and the second object model including a plurality of data objects in a non-eXtensible Markup Language (non-XML);

storing the first and second object models in one or more memories;

generating mapping rules, the mapping rules including XML based syntax that uses rule specification guide to facilitate transforming the one or more data objects in XML to non-XML;

generating a plurality of transformation classes using the first and second object models and the transformation mapping rules, the one or more transformation classes being configured to transform the one or more data objects in XML to non-XML;

receiving one or more data objects;

converting the received one or more data objects, via the transformation classes, (1) in XML to non-XML; or (2) in non-XML to XML; and

transmitting the converted one or more data objects.

12. The method according to claim 11, wherein each of the one or more data objects includes a Java object.

13. The method according to claim 11, wherein the XML includes a domain object model type and the non-XML includes an application-specific object model type.

14. (Cancelled)

15. The method according to claim 11, wherein the one or more data objects are received from a messaging middleware.

16. The method according to claim 11, wherein the one or more data objects are received from an application, the application being coupled to a messaging middleware.

17. A system for data object transformation, the system comprising:

one or more processors;

one or more memories coupled to the one or more processors; and

a data transformation adapter having program instructions stored in the one or more memories, the one or more processors being operable to execute the program instructions, the program instructions including:

receiving a message from a messaging middleware, the message including one or more data objects in an eXtensible Markup Language (XML), wherein the message is in a first communications format;

converting the message from the first communications format to a second communications format;

converting the one or more data objects in XML to a non-eXtensible Markup Language (non-XML), wherein the one or more data objects are converted using a first set of one or more transformation classes, the one or more transformation classes being configured to transform the one or more data objects in XML to non-XML, each of the one or more transformation classes generated using mapping rules, the mapping rules including XML based syntax that uses rule specification guide to facilitate transforming the one or more data objects in XML to non-XML; and

transmitting the one or more data objects in non-XML to an application.

18. The system according to claim 17, wherein first communications format includes a middleware-dependent format, and the second communications format includes a middleware-independent format.

19. The system according to claim 17, wherein each of the one or more data objects includes a Java object.

20. The system according to claim 17, wherein XML includes a domain object model type and the non-XML includes an application-specific object model type.

21. The system according to claim 17, wherein the program instructions further include:

receiving a second message from the application, the second message including one or more data objects in non-XML;

converting the one or more data objects in non-XML to XML, wherein the one or more data objects are converted using a second set of one or more of the transformation classes;

generating a communications line dependent message, the communications line dependent message including the one or more data objects in XML; and

transmitting the communications line dependent message to the messaging middleware.

22. A system for data object transformation, the system comprising:

a communications line;

a computer readable medium executable on a computing system, the computing system coupled to the communications line, the computer readable medium having a transformation adapter, the transformation adapter including:

an assembly/disassembly layer configured to convert messages from a first communications format to a second communications format;

a transformation layer configured to convert data objects in an eXtensible Markup Language (XML) to a non-eXtensible Markup Language (non-XML) using one or more transformation classes, the one or more transformation classes

being configured to transform the one or more data objects in XML to non-XML; and

a method invocation layer;

a transformation class generator coupled to the transformation adapter, the transformation class generator configured to generate the one or more transformation classes using transformation mapping rules, the mapping rules including XML based syntax that uses rule specification guide to facilitate transforming the one or more data objects in XML to non-XML; and

an application coupled to the transformation adapter, wherein the application transmits data to and receives data from the method invocation layer.

23. The system according to claim 22, wherein the communications line includes messaging middleware.

24. The system according to claim 22, wherein each of the one or more data objects includes a Java object.

25. The system according to claim 22, wherein the XML includes a domain object model type and the non-XML includes an application-specific object model type.

26. An apparatus for data object transformation, the apparatus comprising:

means for generating a first object model and a second object model, the first object model including a plurality of data objects in an eXtensible Markup Language (XML), and the second object model including a plurality of data objects in a non-eXtensible Markup Language (non-XML);

means for storing the first and second object models;

means for generating transformation mapping rules, the mapping rules including XML based syntax that uses rule specification guide to facilitate transforming the one or more data objects in XML to non-XML;

means for generating a plurality of transformation classes using the first and second object models and the transformation mapping rules, the one or more transformation classes being configured to transform the one or more data objects in XML to non-XML;

means for receiving one or more data objects;

means for converting the received data objects, via the transformation classes, in XML to non-XML; and

means for transmitting the converted one or more data objects.

VIII. EVIDENCE – APPENDIX

None.

IX. RELATED PROCEEDINGS – APPENDIX

None.